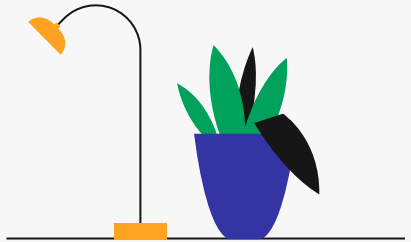
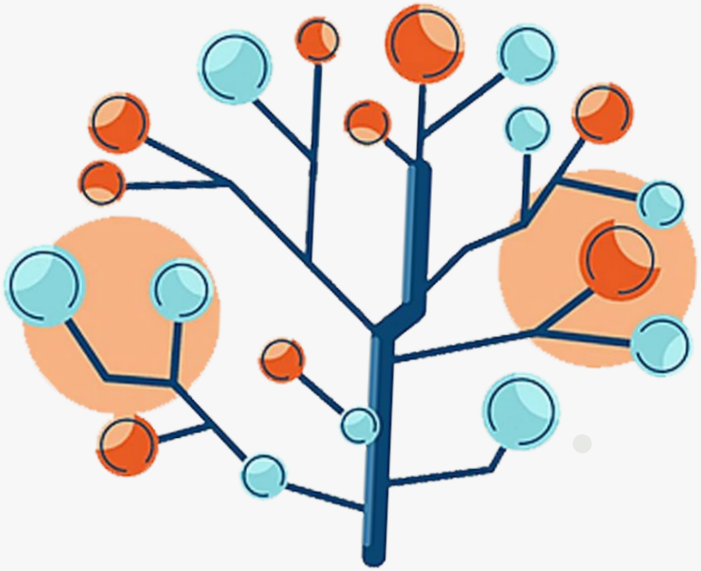
RADIX TREE

EDAA 23/24

João Pinheiro - up202008133
João Oliveira - up202004407
Ricardo Cavalheiro - up202005103

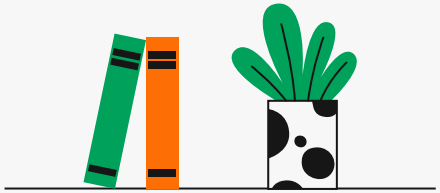
Content





01

What is a Radix Tree?



Data Structure

Trie Tree

A Radix Tree is a compressed Trie (Prefix Tree)

Leaves

The path from the root node to any leaf corresponds to a string

Radix

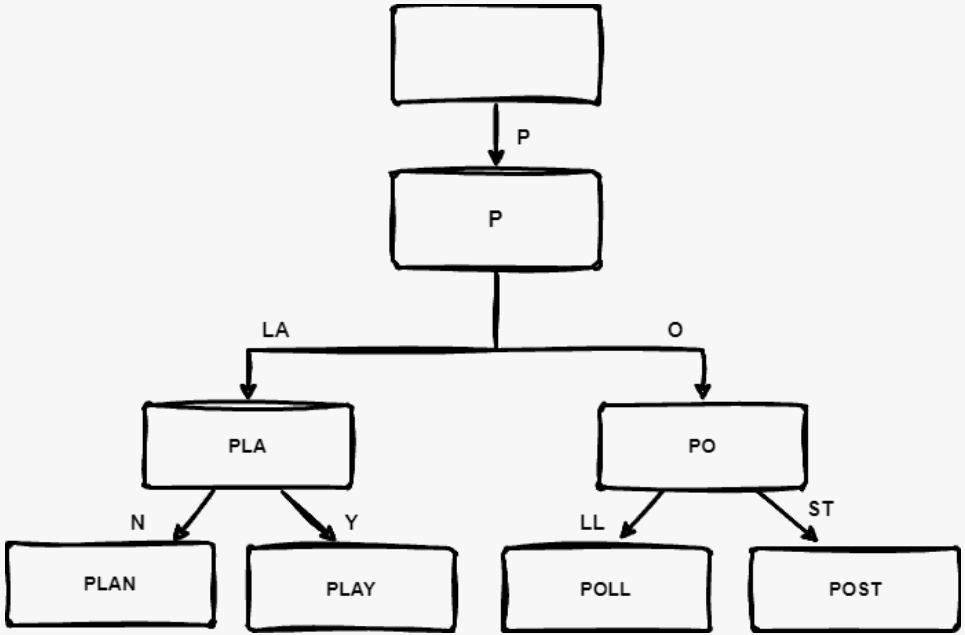
Number of children of every internal node is <= *R* (Power of 2)

Operations

Insertion, Deletion, Search & Common Prefix Strings

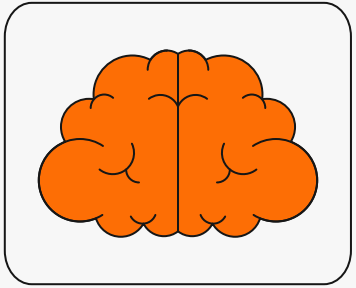
Radix Tree Example

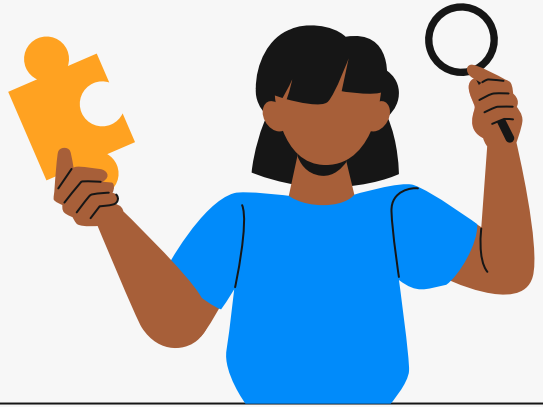
PLAN
PLAY
POLL
POST



02

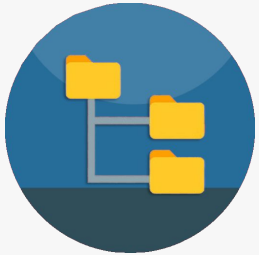
Basic Applications



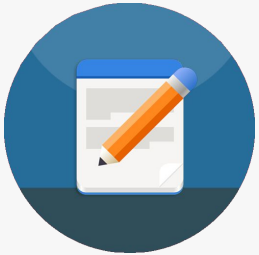


Where & how is it being used

Store IP Routing Tables



Represent & Search File Paths

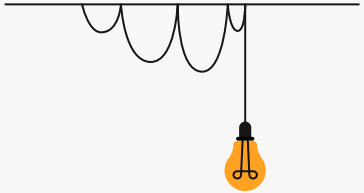


Spell-Checking & Autocomplete

Inverted Indexes

03

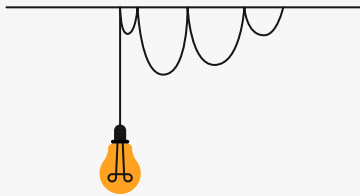
Data Structure Operations



Insertion ⊕

Insert “Hello”

Insert “Hey”



HELLO

HEY

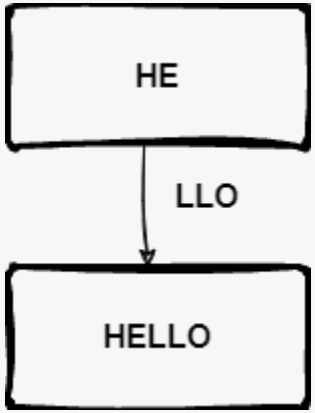
Comparing next letter in insert term to next letter in prefix of current node

Insert “Hey”

HELLO

HEY

Connect new node to the old, and reset prefix stored at the previous node

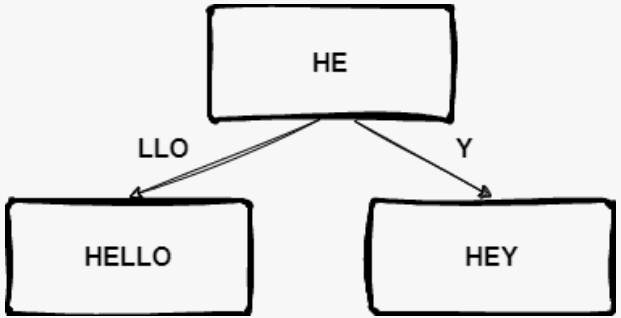


Insert “Hey”

HELLO

HEY

Inserting “Y”

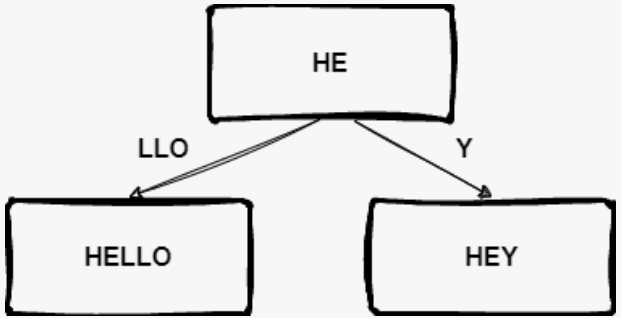


Insert “Rir”

RIR

HE

Comparing next letter in insert term to next letter in prefix of current node



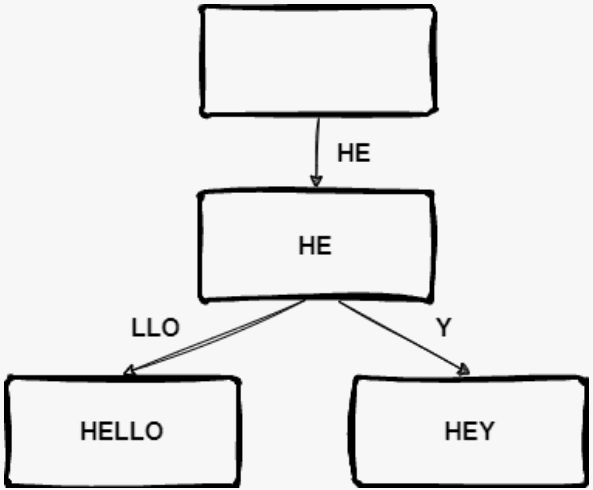
Insert “Rir”

RIR

HE

Prefix mismatch

Create new root node

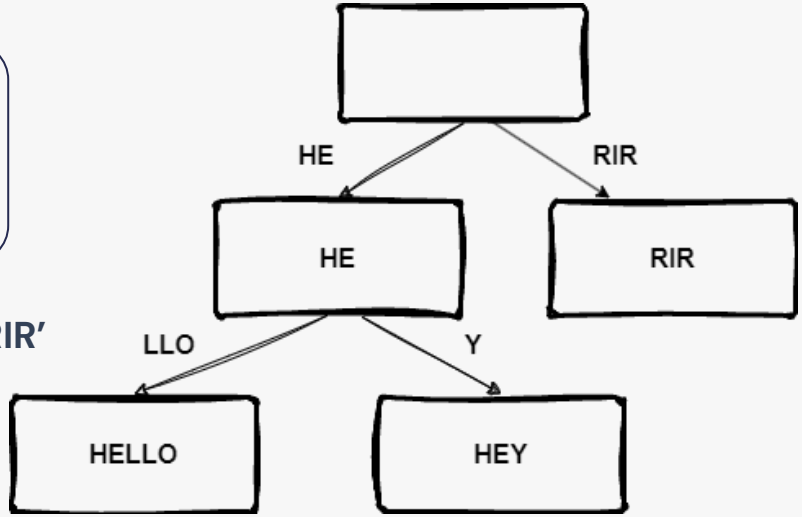


Insert “Rir”

RIR

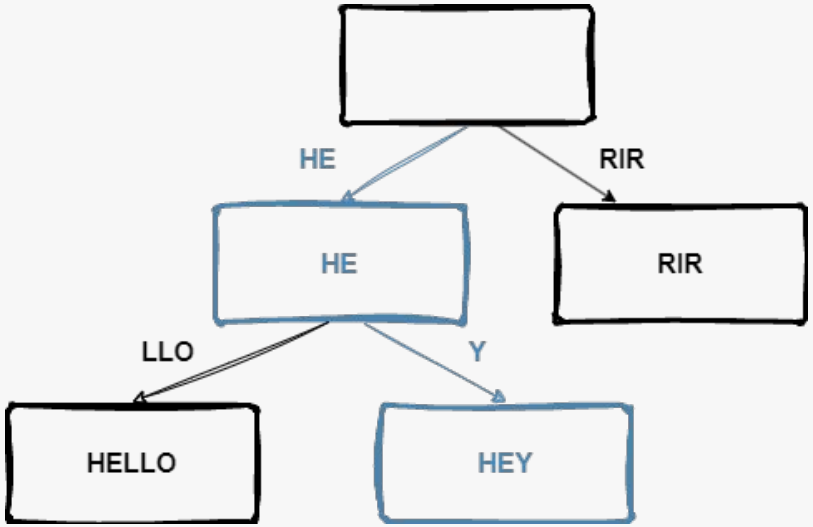
HE

Insert new node ‘RIR’



Search 

Search “Hey”

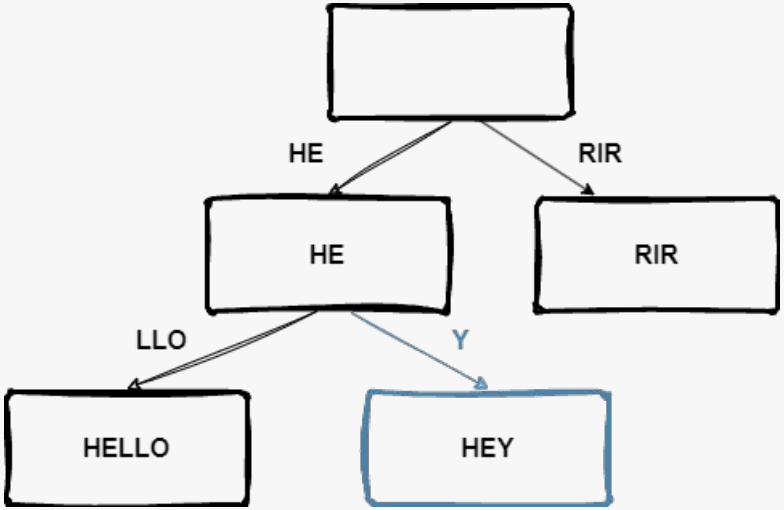


Delete 

Delete “Hey”

Search for “Hey”

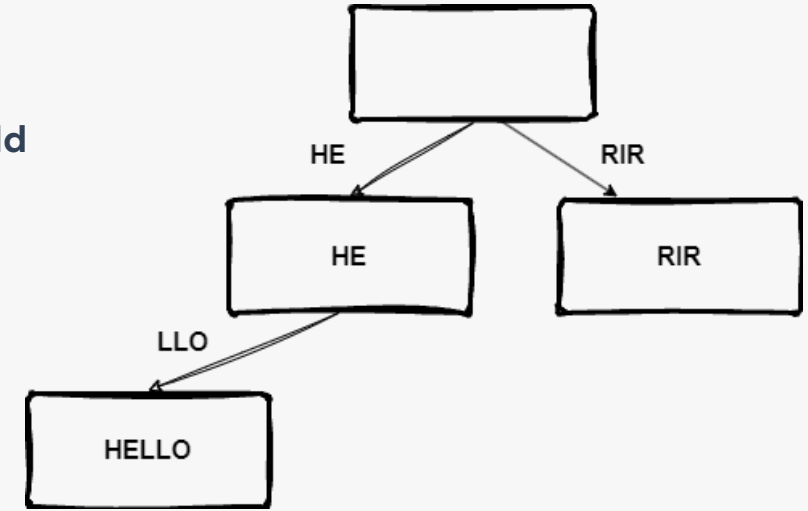
Delete leaf node



Delete “Hey”

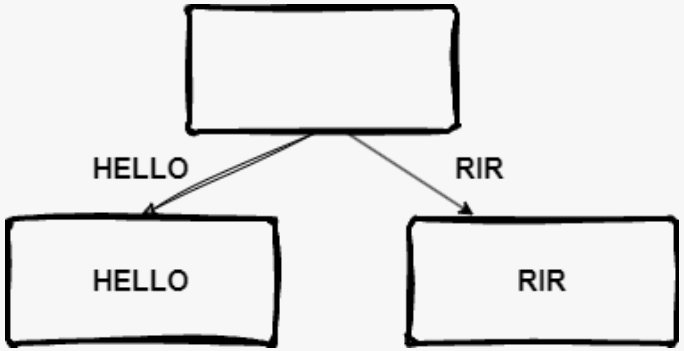
Deleted node

Node left with 1 child



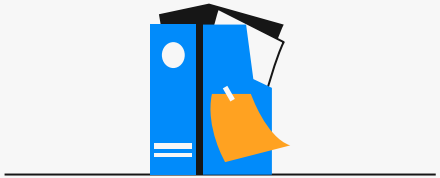
Delete “Hey”

Merged with parent node



04

Addressed Problem



Problems...

Memory Inefficiency

Search Inefficiency

Pattern Matching

Lack of Scalability

05

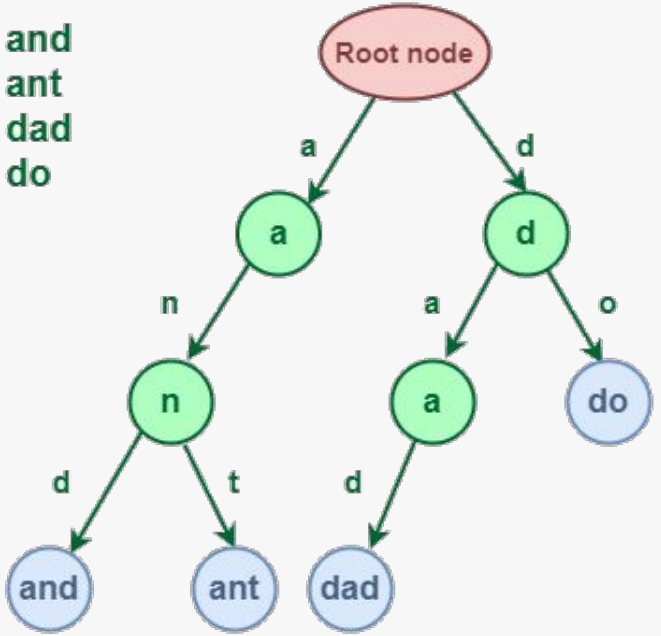
Competing Data Structures

What are the other options?

Trie Tree

Same logic of the radix tree, however each edge is a single character

- and
- ant
- dad
- do

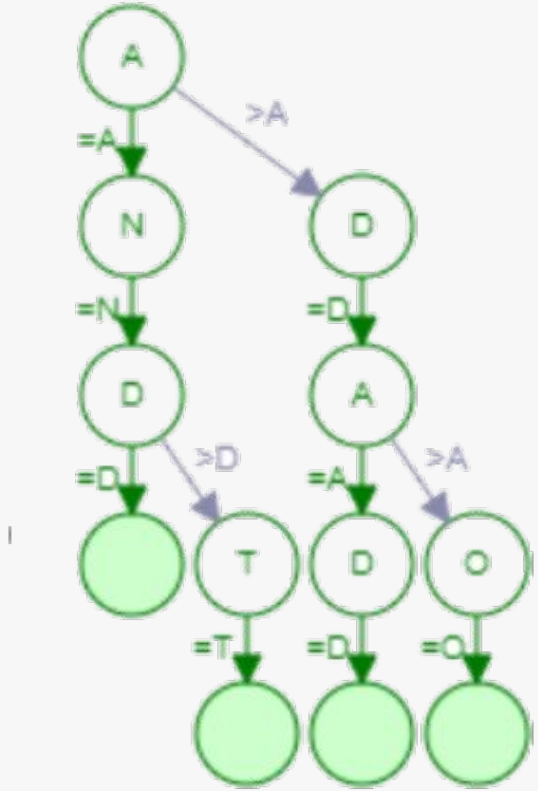


What are the other options?

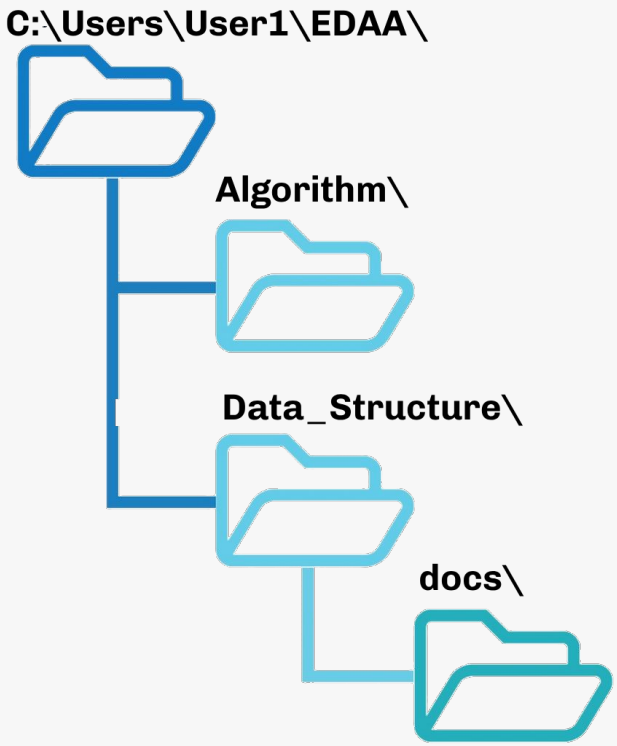
Ternary Search Tree

Each node stores a single character and may have 3 children (<, =, >)

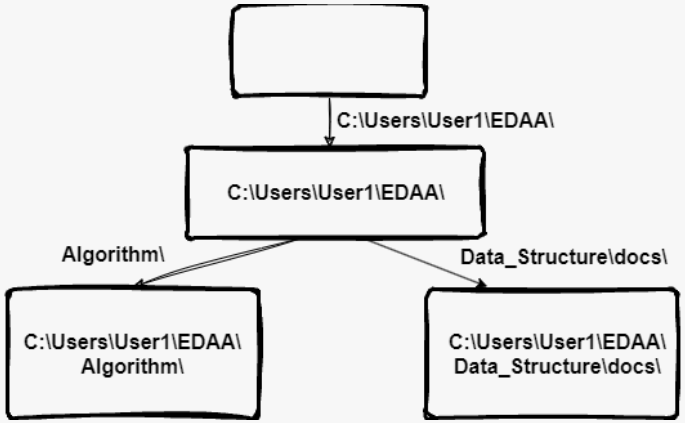
- and
- ant
- dad
- do



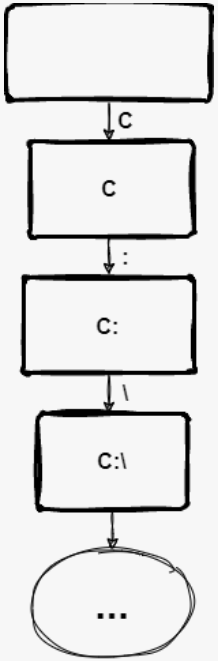
Directory Example...



Directory Example...



Radix Tree



Trie & Ternary Search Tree

Time Complexity

Operation	Radix	Trie	Ternary
Search	O(N)	O(N)	O(log N)
Insertion	O(N)	O(N)	O(log N)
Deletion	O(N)	O(N)	O(log N)

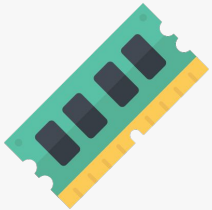
~ Where N is the length of the input string

Conclusions

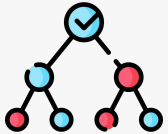
No Rebalancing

Lexicographical Order

Complexity depends on Key Length



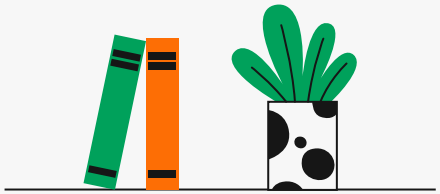
Space Efficiency



Better for non-sparse datasets

06

References



References

Radix Tree
https://medium.com/swlh/ds-algo-problems-tries-and-radix-trees-44df86d8aaf9
https://www.cs.usfca.edu/~galles/visualization/RadixTree.html
https://www.cs.yale.edu/homes/aspnes/pinewiki/RadixSearch.html?highlight=%28CategoryAlgorithmNotes%29

Trie Tree
https://www.cise.ufl.edu/~sahni/dsaac/enrich/c16/tries.htm
https://www.cs.cmu.edu/~avrim/451f11/recitations/rec0921.pdf
https://towardsdatascience.com/finding-all-words-inside-a-string-using-a-trie-prefix-tree-b721ea2f8b6f

Ternary Search Tree
https://www.geeksforgeeks.org/ternary-search-tree/
https://www.youtube.com/watch?v=CIGyewO7868